Performance analysis of concurrent systems with early evaluation*

Jorge Júlvez†
Universitat Politècnica de Catalunya
Barcelona, Spain

Jordi Cortadella
Universitat Politècnica de Catalunya
Barcelona, Spain

Michael Kishinevsky
SCL
Intel Corporation
Hillsboro, USA

ABSTRACT

Early evaluation allows to execute operations when enough information at the inputs has been received to determine the value at the outputs. Systems that can tolerate variable-latency units, such as latency-insensitive or asynchronous systems, can enhance their performance by using early evaluation. The most relevant example of a unit with early evaluation is the multiplexor: the output can be determined as soon as the information of the selected channel arrives, without waiting for the other channels.

This paper analyzes the potential impact of early evaluation in concurrent systems. An analytical model, based on a Petri net extension with early firing is proposed to estimate the performance. The reduction of the analytical model to a linear programming formulation for an efficient estimation of the upper bound for the system throughput is proposed. The results show the accuracy of the model and the benefits of early evaluation.

1. INTRODUCTION

Lazy and eager evaluation are two different strategies to compute the value of expressions. Lazy evaluation is typically used to minimize resource utilization, since expressions are only evaluated when it is strictly necessary. On the other hand, eager evaluation aims at speeding-up computations by evaluating expressions as soon as the values of the variables are available, thus triggering the evaluation of other expressions.

The main motivation of this work comes from the area of latency-insensitive (LI) and asynchronous systems [4, 11]. The computational model of this type of systems resembles dataflow computing. Every wire and storage unit has an extra bit that indicates whether the contents of the component is valid or not. Every computational unit produces a valid result when all input data are valid.

The dynamic behavior of these systems can be often modeled by marked graphs [5], a subclass of Petri nets [9] without choices. Data items are modeled as tokens on the arcs of the graph. When a computational unit has tokens in all input arcs, it can consume all input data and produce a result on the output arcs.

*This work has been partially supported by a gift from Intel Corp. and CICYT TIN 2004-07925.
†Supported by the Spanish Ministry of Education and Science (Juan de la Cierva fellowship).

ICCAD '06, November 5-9, San Jose, CA, USA

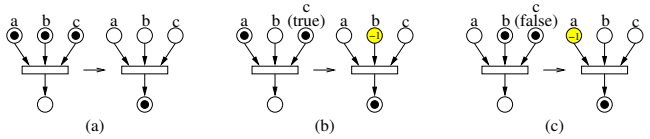


Figure 1: Multi-guarded transitions: (a) AND-causality; (b) early firing with guard $\{a,c\}$; (c) early firing with guard $\{b,c\}$.

1.1 Early evaluation

The requirement that all input data must be available to compute a result is too strict in some cases. For example, if a functional unit computes $a = b * c$, it is not necessary to wait for both operands if one of them is already available and known to be zero. Therefore, the result $a = 0$ could be produced by an *early evaluation* of the expression. A typical component in digital circuits is the multiplexor. The simplest multiplexor has two input data (a and b), one input control signal (c) and one output data (z), with the following behavior:

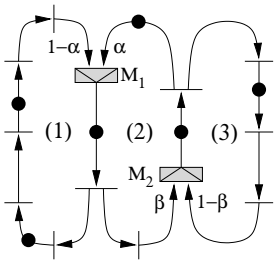
$$z = \texttt{if } c \texttt{ then } a \texttt{ else } b$$

The early evaluation of the multiplexor can be produced if, for instance, c and a are available and the value of c is *true*. In that case, the result $z = a$ can be produced and the value of b can be discarded when it arrives at the multiplexor. Early evaluation has been proposed and used in asynchronous design [2, 10].

Petri nets are not capable of modeling early evaluation, since the enabling of transitions is based on AND-causality, i.e., all input conditions must be asserted. Causal Logic Nets from [12] extend Petri nets to allow transition enabling triggered by arbitrary logic guards associated with transitions. In this paper we present a new model of nets, called *multi-guarded nets* (GN), with the power of modeling early evaluation that associate with a single transition multiple logic guards selected non-deterministically. This non-deterministic selection models interaction of the control with conditions in the data-path.

Figure 1(a) illustrates the firing rule in Petri nets (AND-causality). Early evaluation is modeled by *multi-guarded* transitions. A guard is a subset of places that can enable a transition. A multi-guarded transition has a set of guards from which one of them is chosen nondeterministically at each firing. For example, a transition modeling the previously described multiplexor would have a set of two guards, $\{\{a,c\},\{b,c\}\}$. The availability of information is determined by the presence of a token in the place. For every token, there is an associated data value. For example, c has a Boolean value (*true* or *false*).

However, GNs only model the control of the system. Data-dependent control is modeled by the nondeterminism that selects the guard. In our example, the selection between $\{a,c\}$ and $\{b,c\}$ is what models the fact that the token associated to c can have the value *true* or *false*.



β \ α	0.02	0.2	0.4	0.6	0.8	0.98
0.02	0.403	0.403	0.403	0.403	0.403	0.403
0.2	0.423	0.423	0.424	0.424	0.425	0.426
0.4	0.429	0.436	0.442	0.447	0.451	0.453
0.6	0.430	0.441	0.454	0.465	0.480	0.487
0.8	0.430	0.443	0.460	0.479	0.507	0.530
0.98	0.430	0.443	0.461	0.488	0.527	0.584

Figure 2: Throughput of a GN with probabilistic guards.

When a transition is *early-enabled*, it can fire by storing a *negative token* (-1) in the input places without tokens. This negative token will be cancelled out when the late *positive token* arrives at the place.

1.2 Performance analysis

Evaluating the performance of choice-free systems has been extensively studied by many authors. The performance is determined by the *minimum cycle mean* [7]. If we assign a delay to each node in the marked graph, the mean weight of a cycle is defined as the total delay of the cycle divided by the number of tokens in the cycle. The cycle with the maximum value is the one that determines the performance of the system. The minimum cycle mean can be efficiently computed in polynomial time [6]. The performance of these systems can be increased when the delay of the critical cycles can be partially hidden by sporadic early evaluations.

Figure 2 depicts a marked graph with three cycles. The shadowed transitions M_1 and M_2 model two multiplexors. Their control signals are assumed not to be critical and are not depicted in the graph. Thus, the two input arcs of the multiplexors model the two input data. Associated to each input arc there is a guard and a real number in the interval $[0,1]$ that indicates the probability for the guard to be selected. Each transition is assumed to have unit delay.

Under a pure Petri net model with AND-causality, the performance of the system would be determined by the most stringent cycle. The throughput Th_i (tokens/transitions) for each cycle is the following:

$$Th_1 = \frac{3}{7} = 0.429 \qquad Th_2 = \frac{3}{5} = 0.6 \qquad Th_3 = \frac{2}{5} = 0.4$$

Hence, the global throughput of the system would be 2/5. By incorporating early evaluation, the throughput can be increased, as shown in the table at the right-hand side of the figure. When β is close to 0, the system throughput tends to 0.4, i.e., it is almost completely determined by cycle (3). On the other hand, as α and β approach 1, the throughput increases and tends to 0.6, i.e., cycle (2) determines the system throughput. In general, the throughput may oscillate between 0.4 and 0.6 depending on the probabilities at each multiplexor.

Note that, as it can be seen in this example, the naive method of computing the throughput of the early evaluation system as a weighted sum of the throughputs of the individual loops is incorrect, since loops may affect each other in a complex interplay.

The speculative execution of branches, the selection of data items from the cache memory (instead of main memory), or the bypasses in the pipeline to avoid stalls, are typical examples of early evaluation schemes to improve the performance of microprocessors. In each of these schemes, a multiplexor-based implementation is involved. The impact of these mechanisms in the performance is strongly related to the probability of the events associated to each multiplexor input, e.g. hit ratio of the cache, probability of branch instructions, etc.

Having analytical models for early evaluation contributes to ease the exploration of different architectural mechanisms and evaluate their impact in performance.

2. MULTI-GUARDED NETS

DEFINITION 1 (GN). *A* Multi-Guarded Net *(GN) is a tuple* $\mathcal{N} = \langle P,T,F,G\rangle$ *where:*

- *P is a finite set of places*
- *T is a finite set of transitions*
- *$F \subseteq (T \times P) \cup (P \times T)$ is the flow relation, and*
- *$G : T \to 2^{2^P}$ assigns a set of guards to every transition, such that the following condition is satisfied. Let us define the* preset *and the* postset *of a transition as ${}^\bullet t = \{p|(p,t) \in F\}$ and $t^\bullet = \{p|(t,p) \in F\}$, respectively. Every transition t is assigned a set of guards $G(t)$, where every guard $g_i \in G(t)$ is a subset of of t's preset, i.e., $g_i \subseteq {}^\bullet t$.*

A classical Petri net is simply a GN in which $G(t) = \{{}^\bullet t\}$, for every $t \in T$ (see, e.g. [9] for a tutorial on Petri nets). In a GN, the transitions can also satisfy the condition $G(t) = \{{}^\bullet t\}$. Such transitions will be called *simple* transitions.

DEFINITION 2 (MARKING). *A marking in a GN is a function $m : P \to \mathbb{Z}$, that assigns an integer $m(p)$ to each place p. A* marked *GN is a tuple $\mathcal{N} = \langle P,T,F,G,m_0\rangle$, where m_0 is the initial marking.*

Note that, unlike Petri nets, GNs can have negative markings. If $m(p) \geq 0$ one says that place p has $m(p)$ positive tokens. Otherwise, place p has $|m(p)|$ negative tokens. Negative tokens account for the activity that must be discarded when arriving at the input of the transition, e.g. discarding the branch target address when a branch is not taken.

DEFINITION 3 (FIRING SEMANTICS). *The dynamic behavior of a marked GN is determined by its firing rules. The execution of a transition t can be described as follows:*

- Guard selection. *A guard $g_i \in G(t)$ is selected nondeterministically first in the initial marking, m_0, and then each time t fires. The guard selection is trivial for simple transitions, since they only have one guard. For non simple transitions any guard $g_i \in G(t)$ can be selected. The selected guard of a transition t is* persistent*, i.e., never changes between the firings of t.*
- Enabling. *If the guard $g_i \in G(t)$ has been selected for the next firing of t, then the transition t becomes enabled when every place $p \in g_i$ has a token ($m(p) > 0$).*
- Firing. *A transition t enabled at marking m can fire leading to another marking m' such that*

$$m'(p) = \begin{cases} m(p)-1 & \text{if } p \in {}^\bullet t \setminus t^\bullet \\ m(p)+1 & \text{if } p \in t^\bullet \setminus {}^\bullet t \\ m(p) & \text{otherwise} \end{cases}$$

Every transition locally changes the marking of the net. Different transitions may fire concurrently and simultaneously (if enough positive tokens are present in their common preset places).

- Single-server semantics. *No multiple-instances of the same transition can fire simultaneously. Therefore, a guard selection is produced for each transition firing.*

The persistence of the guards is an accurate abstraction of the conditions for early evaluations (e.g. mux select signals) using non-determinism. The single-server semantics is an abstraction for those systems that communicate through channels using FIFOs.

DEFINITION 4 (REACHABILITY). *The set of reachable markings, R, of a given GN is the set of markings that can be generated from the initial marking, m_0, by iteratively applying the firing rules.*

Note that, in general, the set of reachable markings can be infinite due to the infinite accumulation of positive or negative tokens.

In this paper, we will only consider the subclass of GNs without choice places. It corresponds to the subclass of *marked graphs* (MG) in Petri nets extended to handle early evaluation with negative marking. This subclass is sufficient for modeling a wide class of systems with early evaluation. It also satisfies some properties that simplify their analysis.

DEFINITION 5 (GMG). *A* Multi-guarded Marked Graph *(GMG) is a strongly connected GN with* $|{}^\bullet p| = |p^\bullet| = 1 \ \forall p \in P$.

Places can be omitted in drawings of GMG as shown in figure 2.

2.1 State equation and place invariants

Let C be the $n \times m$ incidence matrix of the GN with rows corresponding to n places and columns to m transitions.

$$C_{ij} = \begin{cases} -1 & \text{if } t_j \in p_i^\bullet \setminus {}^\bullet p_i \\ +1 & \text{if } t_j \in {}^\bullet p_i \setminus p_i^\bullet \\ 0 & \text{otherwise} \end{cases}$$

DEFINITION 6 (P-INVARIANT). *A P-invariant or place invariant is a vector, s, of n nonnegative integers such that*

$$s \cdot C = 0 \tag{1}$$

Similarly to classical MGs [9], it can be shown that every minimal P-invariant of a GMG corresponds to a minimal cycle. Intuitively, an invariant represents a set of places in which the weighted sum of tokens is invariant in any reachable marking.

In the GMG shown in figure 2, the three minimal P-invariants correspond to the three cycles. The total sum of tokens (positive-negative) is invariant for each cycle: 3 for the left and middle loops, and to 2 for the right loop.

As in classical Petri nets, the state equation

$$m = m_0 + C \cdot \sigma \tag{2}$$

gives a necessary condition for the reachability of marking m, where σ is the firing count vector: the j's component corresponds to the number of times transition t_j has fired.

2.2 Timed GMG

In Petri nets, transitions usually represent the events or actions of the system. To model latencies or delays of the system events, a positive real number $\delta(t)$ is associated with every transition t of the GMG. For modeling synchronous systems $\delta(t)$ is typically restricted to be a positive integer (or rational) indicating the number of clock cycles it takes to complete the action. Notice that the firing of GMG transitions is persistent, i.e., a transition cannot be disabled by the firing of another transition. Therefore, a transition t that becomes enabled at time τ will fire at time $\tau + \delta(t)$. It can be shown that more general delay models are possible for the GMG, most notably it is possible to have probabilistic distributions of delays associated with every transition to model variable latency units or delay variations. This however goes beyond the scope of this paper.

The performance evaluation of the model requires some a priori information about probabilities for selecting guards. These probabilities can be obtained from statistical analysis or simulation. It is assumed here that selection of guards for different transitions are independent events and hence probabilities of the guards of different transitions are uncorrelated.

DEFINITION 7 (TGMG). *A* Timed Multi-Guarded Marked Graph *(TGMG) is a tuple* $\mathcal{N} = \langle P, T, F, G, m_0, \delta, \alpha \rangle$ *where:*

- $\langle P, T, F, G, m_0 \rangle$ *is a marked GMG.*
- $\delta : T \to \Re^+ \cup 0$ *assigns a nonnegative delay to every transition.*
- α *is a function that assigns a strictly positive probability to each guard such that for every guarded transition t:* $\sum_{g \in G(t)} \alpha(g) = 1$.

The firing semantics of the TGMG is derived from the previously described semantics of the GMG. It slightly differs in a few aspects:

- Guard selection for every transition is still non-deterministic, but respects probabilities in the infinite executions.
- Firing of transition t takes $\delta(t)$ time units, from the time it becomes enabled until the firing is completed. With the single-server semantics, no multiple instances of the same transition can be firing simultaneously.

DEFINITION 8 (STEADY STATE THROUGHPUT). *The steady state throughput,* $Th(\mathcal{N})$*, of a TGMG is defined as:*

$$Th(\mathcal{N}) = lim_{\tau \to \infty} \frac{\sigma(\tau)}{\tau}$$

where τ *represents the time and* $\sigma(\tau)$ *is the firing count vector at time* τ.

Some authors say that the firing process is "weakly ergodic" if the above limit exists [3]. In the next section, we show that the above limit exists for any TGMG. Note that $Th(\mathcal{N})$ is defined as a vector. Potentially, different transitions, represented by components of $Th(\mathcal{N})$, can have different throughputs. We will show later that in any TGMG all transitions have the same throughput.

2.3 Properties of GMG

DEFINITION 9 (BOUNDEDNESS). *A place, p, of a GN is said to be* bounded *if there exist two integer numbers, l and u, such that* $l \le m(p) \le u$ *for every reachable marking m. A place is k-bounded if* $-k \le m(p) \le k$. *A GN is* bounded *if all its places are bounded.*

A special case of k-boundedness is 1-boundedness (also called safeness) when no more than one positive or negative token can be held in a place.

DEFINITION 10 (DEADLOCK-FREEDOM AND LIVENESS). *A GN is* deadlock-free *iff for every reachable marking m there exists a transition t enabled at m. A GN is* live *iff for every transition t and every reachable marking m, there exists a reachable marking* m' *from m such that t is enabled in* m'.

The following two properties hold for GMGs.

PROPERTY 1. *A GMG is live iff it is deadlock-free.*

Proof:
($\Rightarrow$)
By definition any live GMG is deadlock-free.
($\Leftarrow$)
Let us assume that there exists a GMG that is deadlock-free but it is not live. So, there exists a reachable marking m at which a given set of transitions are deadlocked but the rest can fire indefinitely. This would mean that the live transitions do never require tokens produced by the deadlocked transitions. This cannot be true since the net is strongly connected and all guards can be selected. □

PROPERTY 2. *A GMG is deadlock-free iff for each P-invariant the sum of markings in its corresponding places is positive.*

Proof:
($\Leftarrow$)
Assume a deadlock is reached, i.e., a marking m has been reached at which every transition is disabled. Clearly, at m every transition has at least one preset place, p, with a non-positive marking, $m(p) \leq 0$. Given that the GMG is a strongly connected GN, such set of non-positive places contains at least one cycle that corresponds to a P-invariant.
($\Rightarrow$)
Assume there is a P-invariant such that the sum of markings in its corresponding places is not positive. Let us denote as Q the cycle that corresponds to such P-invariant. Since any guard can be selected with a non-zero probability, a marking m will be eventually reached in which every transition that belongs to the cycle Q selects a guard with a place in Q. Therefore, in m every transition $t \in Q$ waits for a positive token in a place, $p \in {}^\bullet t$, contained in Q (i.e., $p \in Q$) in order to get enabled. Let us assume for a moment that every transition $t \in Q$ can eventually fire from m. This would mean that a positive token has travelled all along Q through the preset places of all the transitions in Q. This would however require the sum of markings in Q to be positive due to the above made assumption that all $t \in Q$ selected guards with places from Q. We have reached a contradiction. Therefore, at least one transition will deadlock forever starting from marking m. By property 1, the rest of the transitions will also eventually deadlock. □

In this paper we are only interested in live GMGs. Liveness of a GMG can be checked in polynomial time using an extension of Commoner's theorem [5]: *a marked graph is live iff every cycle is marked.*

Let us consider the TGMG in figure 3. The guards of transition t_1 are p_a with probability α and p_b with probability $1-\alpha$. Let us first assume that the delays of both transitions are equal to 1. In the initial marking $(pa, pb, pc) = (1,1,0)$, t_1 is enabled and will fire leading to marking $(1,0,1)$. In this marking, t_2 is enabled, while t_1 is enabled with probability α or disabled with probability $1-\alpha$. Figure 3 (left-bottom) shows a reachability graph with the probabilities of transitions between markings. The process of transitioning between markings in the TGMG can be described with a Markov chain whose transition graph corresponds to the reachability graph of the TGMG.

Let us change the delay of transition t_2 and assume that $\delta(t_2) = 2$ and $\delta(t_1) = 1$. A surprising property of GMGs is that, although all the places are covered by P-invariants, the set of reachable markings can be infinite since the net is not bounded. The graph on the right of the figure 3 represents the potential evolutions of the marking for places p_b and p_c (p_a is always 1). Each transition arc corresponds to one time unit. From the initial marking $(pb, pc) = (1,0)$, t_1 fires in 1 time unit leading to marking $(0,1)$. In this marking, t_2 is enabled whereas t_1 is enabled with probability α (in case p_a is selected as guard). Regardless the selection of t_1's guard, t_2 will complete the firing in 2 time units, thus leading to the initial state in case p_b is selected (with probability $1-\alpha$) or leading to the state $(-1,2)$ in case p_a is selected (with probability α). Although, the larger the marking of p_c the smaller the probability of being reached, it is not possible to define neither an upper bound for p_c nor a lower bound for p_b.

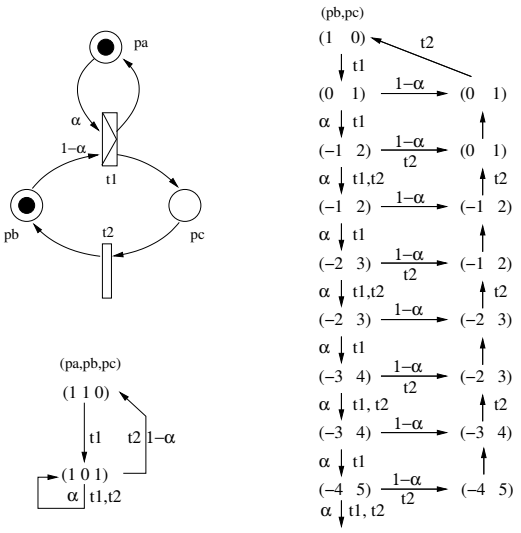


Figure 3: A TGMG and its Markov chain for the bounded ($\delta(t_1) = \delta(t_2) = 1$) and unbounded case ($\delta(t_1) = 1, \delta(t_2) = 2$).

The following theorem states however that the unique steady state throughput exists even for the unbounded TGMGs.

THEOREM 3. *The limit $lim_{\tau\to\infty}\frac{\sigma(\tau)}{\tau}$ exists for any TGMG.*

Proof: If the net is not live it will deadlock (property 1) and hence the steady state throughput of all transitions will be 0. Otherwise, for a given transition t, the expression $\frac{\sigma_\Delta}{\Delta}$ (where σ_Δ is the firing count vector during interval Δ) is upper bounded by $\frac{1}{\delta(t)}$ and lower bounded by 0 for every $\Delta > 0$. The limit would not exist if $\frac{\sigma_\tau}{\tau}$ could oscillate indefinitely between two values u and l, where $u > l$. Let us assume that such an oscillation is possible. Thus, during a given time interval Δ_1, the selected guards make $\frac{\sigma_{\Delta_1}}{\Delta_1}$ be greater than u, and during a given time interval Δ_2, the selected guards make $\frac{\sigma_{\Delta_2}}{\Delta_2}$ be less than l. Since $\frac{\sigma_\Delta}{\Delta}$ is upper and lower bounded for any $\Delta > 0$, the only way to keep the oscillations between u and l is by growing the intervals Δ_1 and Δ_2 as time passes. However, the probability of selecting the guards that make $\frac{\sigma_{\Delta_1}}{\Delta_1} > u$ tends to 0 as Δ_1 increases. Hence, the probability of oscillating indefinitely is 0. □

As in classical MGs, the steady state throughput of a TGMG can be characterized by a single scalar number.

THEOREM 4. *The steady state throughput of all transitions of a TGMG is the same.*

Proof: Let us assume that two transitions t_1 and t_2 exist such that $Th(t_1) < Th(t_2)$. This would mean that in the steady state t_2 does not require tokens produced by t_1, and in turn, the tokens produced by the transitions that are in a directed path that starts at t_1. This implies that either t_1 and t_2 are not connected or some guards have probability 0. Contradiction. □

This fact eases the performance evaluation of a TGMG: to compute the throughput of the system it is enough to compute the throughput of one transition. Notice that theorem 4 would not hold if some guard probabilities are equal to zero. Guards with zero probability might result in disconnected graph components with different throughputs. As an example, consider $\alpha = \beta = 0$ in the example of figure 2: Cycles (1) and (3) would become disconnected with throughputs $3/7$ and $2/5$ respectively.

Theorem 4 indicates that all transitions of a TGMG fire, on average, the same number of times in an infinite execution of the system. Therefore, the evolution of the marking will exhibit a repet-

itive behavior, what implies the existence of average marking of every place.

THEOREM 5. *For every TGMG, there exists a limit*

$$\overline{m} = lim_{\tau\to\infty}\frac{1}{\tau}\int_0^{\tau} m(\xi)d\xi$$

where $\overline{m}$ is a vector with components $\overline{m}(p)$ representing the average marking of individual places.

3. ANALYSIS WITH MARKOV CHAINS

Due to the stochastic nature of selecting guards a TGMG can be viewed as a continuous time stochastic process. As we have mentioned before the evolution of the TGMG therefore can be expressed as a continuous time Markov chain with a transition graph isomorphic to the reachability graph of the TGMG. Deriving a Markov chain out of the TGMG is a variation of standard for the stochastic Petri Nets technique [1,8] and therefore is illustrated here by example.

Figure 4 shows a TGMG (delays of all transitions assumed to be 1) and the associated transition graph of a Markov chain. Each arc in the Markov chain corresponds to one time unit. For simplicity, the transitions of the TGMG are named a,b,c,d and places ab,ba,ac,cd,da using pairs of names of preset-postset transitions. The states of this graph S_1,S_2,S_3 are the reachable markings. The matrix-like shape depicted at each state correspond to the marking at each state (see graphical explanation in figure 4). Arcs are labeled with probabilities to be taken (omitted if probability is 1) and a set of firing transitions.

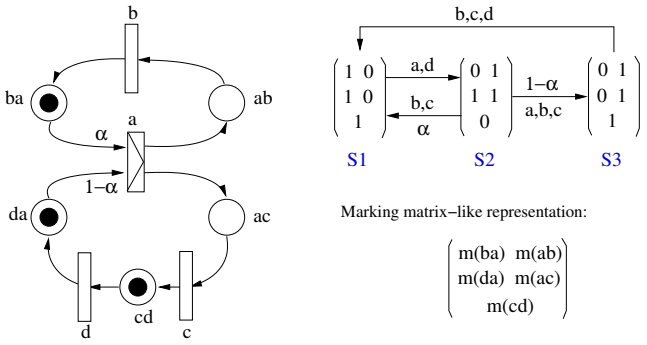


Figure 4: A TGMG and its associated Markov chain.

The average time spent at each state (marking) at the steady state can be obtained by solving the set of linear equations corresponding to the Markov chain. Let Z_1,Z_2,Z_3 be the probabilities to be in the corresponding states S_1,S_2,S_3 during the steady state. One can write a set of equations corresponding to the transitions of the Markov chain:

$$\begin{aligned} Z_2 &= Z_1 \\ Z_3 &= (1-\alpha)\cdot Z_2 \\ Z_1+Z_2+Z_3 &= 1 \end{aligned}$$

The solution is:

$$Z_1 = Z_2 = \frac{1}{3-\alpha}\ ,Z_3 = \frac{1-\alpha}{3-\alpha}$$

Transition a is fired with probability 1 from S_1 and with probability $1-\alpha$ from S_2. Therefore, the steady state throughput of transition a is:

$$Th(a) = Z_1 + (1-\alpha)\cdot Z_2 = \frac{2-\alpha}{3-\alpha}$$

Theorem 4 states that the rest of transitions have the same throughput. Notice that the computed throughput tends to $1/2$ (the throughput of the upper cycle) as α approaches 1, and to $2/3$ (the throughput of the lower cycle) as α approaches 0.

For a standard MG, it is known that the throughput can be obtained as a minimal throughput of its cycles, or equivalently as an inverse to the maximal cycle time among all the cycles. It is important to note that extending either of these techniques to the TGMG by averaging the cycle throughputs (or cycle times) does not yield correct results. Computing the throughput as an average of individual cycle throughputs gives:

$$\frac{1}{2}\cdot\alpha+\frac{2}{3}\cdot(1-\alpha) = \frac{4-\alpha}{6}$$

Computing it as an inverse to an average cycle time gives:

$$\frac{1}{2\cdot\alpha+\frac{3}{2}\cdot(1-\alpha)} = \frac{2}{3+\alpha}$$

Neither of these expressions is equal to the exact throughput expression above computed with Markov chains.

The use of Markov chains allows one to compute the exact throughput of any bounded TGMG. However, it requires an exhaustive exploration of the reachability graph that is exponentially larger than the size of the bounded TGMG.

4. A BOUND ON TGMG THROUGHPUT

Let us first consider the sub-class of TGMG in which all guards are singleton sets, i.e., each guard contains only one place: $g \in G(t)\ |g| = 1$. All examples considered in this paper so far, except the one in figure 1, belong to this sub-class. Nets from this class are easier for analysis since tokens within guards do not interfere with each other. We will later show how to transform any TGMG to an equivalent singleton form.

Let t be a transition of a TGMG, $\delta(t)$ its delay, and $prob(en(t))$ be the probability of t being enabled during the steady state computation. $prob(en(t))$ can be thought as the average proportion of time during which t is enabled. Since transitions have deterministic delays and operate under the single server semantics, the following equation holds:

$$\delta(t)\cdot Th(t) = prob(en(t)) \tag{3}$$

Let us denote the average marking of place p in the steady state as $\overline{m}(p)$. Equation 3 can be used to obtain the following theorem that gives an upper bound for the steady state throughput of a TGMG with singleton guards and simple transitions.

THEOREM 6. *Let $\mathcal{N}$ be a TGMG such that every transition t is simple or has singleton guards.*

(A) Let t be a guarded transition with singleton guards. Let $\alpha(p)$ be the probability of the singleton guard $g=\{p\}\in G(t)$. Then:

$$Th(t) \le \frac{\sum_{p\in\bullet t}\alpha(p)\cdot\overline{m}(p)}{\delta(t)}$$

(B) Let t be a simple transition. Then:

$$Th(t) \le \frac{\min_{p\in\bullet t}\overline{m}(p)}{\delta(t)}$$

See the Appendix for a sketch of the proof of theorem 6. For the following two sub-classes the throughput bound of theorem 6 is the exact system throughput.

COROLLARY 7. *(A) Let $\mathcal{N}$ be a 1-bounded TGMG with singleton guards for all its transitions. Let t be one of the transitions, then:*

$$Th(t) = \frac{\sum_{p\in\bullet t}\alpha(p)\cdot\overline{m}(p)}{\delta(t)}$$

(B) Let $\mathcal{N}$ be a 1-bounded TGMG with simple transitions only (i.e., a timed MG). Let t be one of the transitions, then:

$$Th(t) = \frac{\min_{p \in {}^\bullet t} \overline{m}(p)}{\delta(t)}$$

Notice that simple transitions with only one preset place are a particular case of transitions with singleton guards. Thus, they fulfil both condition (A) and (B) of corollary 7.

4.1 Reduction to singleton form

Figure 5 shows a fragment of a TGMG with a guarded transition t whose guards $G(t) = \{\{a,b\},\{b,c\}\}$ are not singletons. An equivalent TGMG has a guarded transition with singleton guards. Two new simple transitions, t_1 and t_2, with zero delay are introduced. They combine together the guards $\{a,b\}$ and $\{b,c\}$ of the original transition t. Note that place b is duplicated. It can be proven that this transformation can transform any TGMG to a singleton form without changing its throughput.

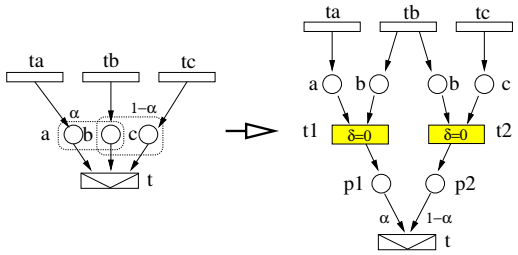


Figure 5: Reduction to singleton form.

5. LP FORMULATION

Let us consider a TGMG such that every transition t is simple or has singleton guards. According to theorem 6, the steady state throughput of each guarded transition t must satisfy the following inequality:

$$\delta(t) \cdot Th(t) \leq \sum_{p \in {}^\bullet t} \alpha(p) \cdot \overline{m}(p)$$

and each simple transition t must fulfill:

$$\delta(t) \cdot Th(t) \leq \overline{m}(p)$$

for every preset place $p \in {}^\bullet t$.

On the other hand, the average steady state marking $\overline{m}$ is necessarily a solution of the state equation:

$$\overline{m} = m_0 + C \cdot \sigma,\ \sigma \geq 0$$

One can combine the above constraints on the throughput and on the average marking, to build a Linear Programming Problem (LP) that maximizes a parameter ϕ, corresponding to the TGMG throughput (one scalar variable suffices since the throughput of all transitions is the same):

$$\begin{aligned} &\textit{Maximize}\ \ \phi: \\ &\delta(t) \cdot \phi \leq \sum_{p \in {}^\bullet t} \alpha(p) \cdot \widehat{m}(p),\ \text{for every } t \in {}^\bullet T_1 \\ &\delta(t) \cdot \phi \leq \widehat{m}(p) \text{ for every } p \in {}^\bullet T_2 \\ &\widehat{m} = m_0 + C \cdot \sigma \\ &\phi \leq \min_{t \in T} 1/\delta(t) \end{aligned} \qquad (4)$$

Where T_1 is the set of transitions with singleton guards, and T_2 is the set of simple transitions. Transitions with only one preset place can be included either in T_1 or in T_2. The vector σ represents the firing count vector that drives the system from the initial marking, m_0, to the estimated average marking $\widehat{m}$. The constraint $\sigma \geq 0$ has been dropped since for any positive σ there are as well partially negative σ's that deliver the same maximal value of ϕ (this is due to the fact that C is not a full rank matrix). The last constraint $\phi \leq \min_{t \in T} 1/\delta(t)$ ensures a single server semantics (i.e., that more than one copy of the same transition cannot fire at the same time). This constraint is redundant for 1-bounded TGMG. It can be shown that the solution of the LP (4) always exists.

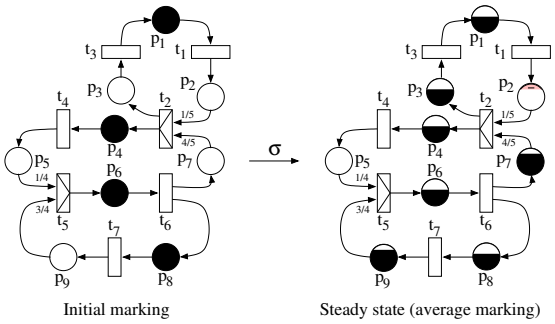


Figure 6: Reaching an average marking

Figure 6 illustrates how the LP problem (4) works on a particular system. Transition t_2 has two guards $\{p_2\}$ and $\{p_5\}$ with probabilities $1/5$ and $4/5$. Transition t_5 has also two guards $\{p_5\}$ and $\{p_9\}$ with probabilities $1/4$ and $3/4$. In the figure, the higher the blank level in a place the higher its occupancy, i.e., a fully black place corresponds to a marking of 1. The estimated average marking given by the LP is $\widehat{m} = (0.6, -0.2, 0.6, 0.6, 0.0, 0.6, 0.8, 0.6, 0.8)$. One of the possible firing count vectors driving the system from m_0 to $\widehat{m}$ is $\sigma = (0.4, 0.6, 0, 1, 1, 1.4, 1.8)$. It is interesting to notice that the average marking of p_2 is negative. However, this does not imply that $m(p_2)$ is always negative in the steady state. In fact, it must become positive to enable transition t_2 when the selected guard is $\{p_2\}$.

The solution of the LP problem (4) returns an upper bound on the throughput for the TGMG.

THEOREM 8. *Let $\mathcal{N}$ be a TGMG with singleton guards. Let ϕ be the solution of LP problem* (4)*. Then the throughput of every transition t satisfies the upper bound: $Th(t) \leq \phi$.*

This theorem enables an efficient method for finding an upper bound on the throughput of a TGMG. The bound can be found in polynomial time (since LP is polynomial). Moreover, corollary 7 defines two sub-classes of TGMG for which the solution of the LP problem is guaranteed to be exact.

Example of LP model

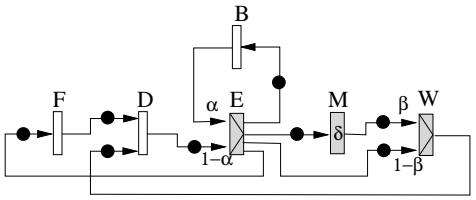
Consider again the 1-bounded TGMG from figure 4 (delays of all transitions assumed to be 1). The associated LP problem is:

Maximize ϕ:

ϕ	$\leq$	ab	for transition b
ϕ	$\leq$	ac	for transition c
ϕ	$\leq$	cd	for transition d
ϕ	$\leq$	$\alpha \cdot ba + (1-\alpha) \cdot da$	for transition a
ba	=	$1 + b - a$	for place ba
da	=	$1 + d - a$	for place da
ab	=	$a - b$	for place ab
ac	=	$a - c$	for place ac
cd	=	$1 + c - d$	for place cd

The solution to this problem is

$$\phi = \frac{2-\alpha}{3-\alpha}$$



	δ = 0.2		δ = 0.4		δ = 0.8	
	β = 0.2	β = 0.3	β = 0.2	β = 0.3	β = 0.2	β = 0.3
α = 0.1	0.823	0.820	0.714	0.714	0.556	0.555
α = 0.2	0.795	0.792	0.712	0.711	0.556	0.555
α = 0.3	0.754	0.751	0.705	0.702	0.556	0.555

Figure 7: Analysis of elastic DLX with early evaluation and variable latency.

which corresponds exactly to the solution we have obtained with Markov chain analysis. The solution obtained by the LP is necessarily the exact throughput since condition (A) of corollary 7 is fulfilled. For $\alpha = 0.5$ the throughput is 0.6 with average marking $\overline{m}(ab) = 0.6, \overline{m}(ac) = 0.6, \overline{m}(cd) = 0.6, \overline{m}(ba) = 0.4, \overline{m}(da) = 0.8$.

6. EXPERIMENTAL RESULTS

6.1 Modeling of an elastic DLX

The paper [4] shows the idea of latency-tolerant, aka synchronous elastic, systems without early evaluation using different versions of a DLX microprocessor. Figure 7 shows a GMG model of an elastic DLX with two early evaluation stages [1]. The model contains transitions named F, D, E, M, W corresponding to the five basic stages of the pipeline: Fetch, Decode, Execute, Memory, and Write-back.

Transition B models a bubble in the bypass network due to an assumed long wire delay. This bubble leads to a stall in the execution when bypass of the result from the previously executed instruction is needed by the next instruction.

The probability of the data dependency between instructions is given by α. The E stage is modeled as a guarded transition that takes results either from the decode stage (probability $1-\alpha$) or the bypass (probability α).

The Write-back stage is modeled by another guarded transition, W, that takes results either from the memory stage in case of executing a Load memory operation or directly from the execution stage in case of other instructions. β stands for the probability of Load instructions.

We also assume that the memory stage, M, has a variable latency. This is typical for a memory sub-system. E.g., if M models only the first level cache, then it would have small latency in case of a hit, and a long latency in case of a miss. The average latency of M is equal to $1+\delta$. The latency of the rest of transitions is 1.

Under a regular MG model with AND-causality (without the guarded transitions), the throughput of the system would be determined by the most stringent bypass cycle and would be equal to 0.5. The throughput for a system with early evaluation is much higher and is captured in the table shown in figure7 for a few values of α, β and δ.

6.2 Random graphs with early evaluation

To illustrate the impact of early evaluation and validate the correctness of the approach for estimating throughput on TGMGs, we performed some experiments on sequential circuits from the MCNC benchmarks.

The circuits were decomposed into 2-input gates and then transformed into TGMGs as follows: (1) The largest strongly connected component from the graph was extracted, (2) each edge was assigned a token with probability 0.75, (3) each node was configured as a 2-input mux with probability 0.25 (i.e., 1 out of 4 nodes was a mux, on average), (3) the probability of each input channel of a mux was generated randomly, (4) all nodes were assumed to have unit delay and (5) complementary arcs were included to guarantee the graph to be 1-bounded (see [9] for details on how to make a graph 1-bounded).

Circuit			Throughput				
Name	**Nodes**	**Edges**	**MG**	**Sim**	**LP**	**ΔTh**	**Err**
s27	22	32	0.333	0.333	0.333	0%	0%
s208	12	15	0.500	0.571	0.594	14%	4%
s298	5434	10040	0.091	0.120	0.129	32%	8%
s349	73	114	0.333	0.333	0.333	0%	0%
s382	28	46	0.250	0.284	0.294	14%	4%
s386	121	204	0.400	0.400	0.400	0%	0%
s400	30	50	0.400	0.438	0.470	10%	1%
s444	34	58	0.200	0.261	0.287	31%	7%
s510	367	671	0.167	0.167	0.167	0%	0%
s526	46	67	0.333	0.333	0.333	0%	0%
s641	89	138	0.333	0.393	0.432	18%	3%
s713	104	167	0.250	0.333	0.333	33%	12%
s820	424	738	0.143	0.201	0.230	41%	7%
s832	474	819	0.286	0.310	0.342	8%	1%
s953	156	259	0.286	0.295	0.333	3%	5%
s1423	396	711	0.100	0.184	0.189	84%	21%
s1488	564	1003	0.188	0.236	0.271	26%	3%
s1494	564	1000	0.154	0.222	0.277	44%	3%
s5378	736	1320	0.235	0.250	0.250	6%	3%
s9234	867	1658	0.200	0.219	0.248	10%	2%

Table 1: Experimental results with sequential circuits.

Table 1 reports the characteristics and the results obtained for the circuits. The column **MG** reports the throughput of the system without early evaluation. The column **LP** reports the upper bound of the estimated throughput with early evaluation using the LP model. Finally, the column **Sim** reports the results obtained by simulation[2].

The first observation is that early evaluation has a tangible impact in the performance of the system. This is reported in the column **ΔTh**, calculated as (Sim-MG)/MG. In one of the examples (`s1423`) the improvement in throughput was 84%. In some other cases, the early evaluation had no impact, due to the fact that the most stringent cycle of the graph had no muxes.

Considering MG and LP as lower and upper bounds, respectively, for the throughput with early evaluation, we report the column **Err** calculated as $|Sim - \widehat{Th}|/Sim$, where $\widehat{Th}$ is throughput estimated as the mid point of the interval [MG,LP]. In most cases, the error was smaller than 5%. Only in a couple of cases in which early evaluation had a significant impact (`s713` and `s1423`), the error was larger. In these cases, the simulation reported a throughput close to the upper bound.

The CPU required to obtain $\widehat{Th}$ using the LP model was always smaller than 2 seconds, and about two orders of magnitude faster than the simulation[3].

[1] A correct modeling of elastic systems, in general, requires representing both forward flow and backward flow (stall propagation) that can be done with pipelined MG. Although it is easy to model the backward flow with GMG, for simplicity, we represent only the forward flow here.

[2] 5000 cycles of simulation were run for each example, guaranteeing results with very small standard deviation.

[3] Only in the largest case, `s298`, the LP model required about 2 minutes to be solved. All the LPs were solved by a PC Pentium IV 2.4 Ghz using the GLPK (GNU Linear Programming Kit).

7. CONCLUSIONS

Tolerance to variable delays not only makes systems more elastic, but opens the opportunity to incorporate new execution schemes. One of these schemes is early evaluation, that can contribute to improve the performance of latency-insensitive and asynchronous systems.

This paper proposes an analytical method to estimate the performance of systems with early evaluation. The method calculates an upper bound of the throughput that is guaranteed to be exact for single-guarded 1-bounded TGMGs and for MGs. The experiments show that it often calculates the exact result even for the general class of the TGMGs.

A surprising fact is that the upper bound can even be calculated for unbounded systems that can accumulate an infinite amount of positive and negative tokens.

The efficiency of the analytical model suggests that it could be used for a fast architectural exploration by detecting the units (e.g. multiplexors) that are the bottlenecks of the system and, thus, candidates to be executed in early-evaluation mode.

8. REFERENCES

[1] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. Wiley, 1995.

[2] C. Brej and J. Garside. Early output logic using anti-tokens. In *Int. Workshop on Logic Synthesis*, pages 302–309, May 2003.

[3] J. Campos, G. Chiola, and M. Silva. Ergodicity and throughput bounds of Petri nets with unique consistent firing count vector. *IEEE Transactions on Software Engineering*, 17(2):117–125, February 1991.

[4] L. Carloni, K. McMillan, and A. Sangiovanni-Vincentelli. Theory of latency-insensitive design. *IEEE Transactions on Computer-Aided Design*, 20(9):1059–1076, Sept. 2001.

[5] F. Commoner, A. W. Holt, S. Even, and A. Pnueli. Marked directed graphs. *Journal of Computer and System Sciences*, 5:511–523, 1971.

[6] A. Dasdan and R. K. Gupta. Faster maximum and minimum mean cycle algorithms for system performance analysis. *IEEE Transactions on Computer-Aided Design*, 17(10):889–899, 1998.

[7] R. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23:309–311, 1978.

[8] M. K. Molloy. Performance Analysis Using Stochastic Petri Nets. *IEEE Trans. on Computers*, 31(9):913–917, 1982.

[9] T. Murata. Petri Nets: Properties, analysis and applications. *Proceedings of the IEEE*, pages 541–580, Apr. 1989.

[10] R. Reese, M. Thornton, C. Traver, and D. Hemmendinger. Early evaluation for performance enhancement in phased logic. *IEEE Transactions on Computer-Aided Design*, 24(4):532–550, Apr. 2005.

[11] J. Sparsø and S. Furber, editors. *Principles of Asynchronous Circuit Design: A Systems Perspective*. Kluwer Academic Publishers, 2001.

[12] A. Yakovlev, M. Kishinevsky, A. Kondratyev, L. Lavagno, and M. Pietkiewicz-Koutny. On the models for asynchronous circuit behaviour with OR causality. *Formal Methods in System Design*, 9(3):189–233, 1996.

APPENDIX

Sketch of the proof of theorem 6

Part A.

Let us consider a small fragment of a TGMG shown in Figure 8. Let the guards of t_1 be $G(t) = \{\{a\},\{b\}\}$ and the probabilities of their selection be α and β. Let us assume that the delays of all transitions are the same and that t_2 is not enabled. Thus, from marking $m(a) = 1$, $m(b) = 0$, two transitions of the corresponding Markov chain are possible:

- Guard $\{a\}$ is selected, transition t_1 is enabled and fired leading to a new marking $m'(a) = 0, m'(b) = -1$ with a negatively marked place b.
- Guard $\{b\}$ is selected, transition t_1 is not enabled and hence the marking remains the same $m'(a) = 1$, $m'(b) = 0$ one time unit later. Due to persistence in guard selection, transition t_1 will not get enabled until t_2 delivers a token into place b.

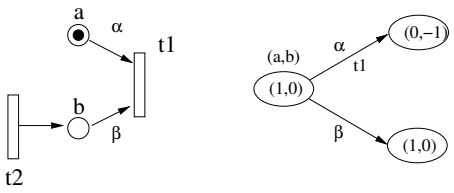


Figure 8: Evolution of guarded transition t_1.

Thus:

$$\frac{\alpha}{\beta} = \frac{prob(m'(a) = 0 \wedge m'(b) = -1)}{prob(m'(a) = 1 \wedge m'(b) = 0)}$$

This reasoning can be applied recursively to the next states $(0,-1)$ and $(1,0)$ implying the following lemma:

LEMMA 9. *Let t be a transition with two singleton guards, then:*

$$\alpha \cdot prob(not\_en_a) = \beta \cdot \overline{m}(b)^-$$

where $prob(not\_en_a)$ is the steady state probability for transition t not to be enabled due to the selection of the guard $\{b\}$ and $m(b) \leq 0$; and $\overline{m}(b)^-$ is the average negative marking of place b, i.e., $\overline{m}(b)^- = prob(m(b) = -1) + 2 \cdot prob(m(b) = -2) + \ldots$

The following theorem can be obtained from lemma 9:

THEOREM 10. *If transition t has two singleton guards, then:*

$$\delta(t) \cdot Th(t) = \alpha \cdot \left(\overline{m}(a) - \sum_{i=2}^{\infty} (i-1) Prob(m(a) = i) \right) + \beta \cdot \left(\overline{m}(b) - \sum_{i=2}^{\infty} (i-1) Prob(m(b) = i) \right)$$

Proof: We first establish the logical expression for $prob(enab(t))$. This expression is transformed into an arithmetic formula that includes terms with $\overline{m}(b)^-$ and $prob(not\_en_a)$. After regrouping the terms of the expression, lemma 9 and equation 3 yield the proof. □

Theorem 10 can be generalized for an arbitrary number of singleton guards and for transitions with different delays. Theorem 6(A) trivially follows from this generalization.

Part B.

Let t be a simple transition. Then:

$$prob(enab(t)) = prob\Big(\bigwedge_{p \in {}^\bullet t} m(p) \geq 1 \Big) \leq \min_{p \in {}^\bullet t} \overline{m}(p)$$

The application of equation 3 yields:

$$Th(t) \leq \frac{\min_{p \in {}^\bullet t} \overline{m}(p)}{\delta(t)}$$